Finite-Time Convergence in Continuous-Time Optimization

A Replication Study

MAE5370 - Optimizations for Engineers

Final Project Report

Trisha Babu

Utah State University

December 2025

Reference Paper:

Romero, O. & Benosman, M. (2020). Finite-Time Convergence in Continuous-Time Optimization. ICML 2020.

Abstract

This report presents a replication study of the numerical experiments from "Finite-Time Convergence in Continuous-Time Optimization" by Romero and Benosman, published at ICML 2020. The paper introduces finite-time convergent gradient flows for optimization, including the q-rescaled gradient flow (q-RGF) and the rescaled Newton flow (RNF). Unlike traditional gradient descent methods that achieve only asymptotic convergence, these flows reach the optimum in a finite, bounded time. This replication study implements the theoretical framework in Python, verifies the key theorems through numerical simulation, and reproduces all five figures from Section 5 of the original paper. The results confirm the finite-time convergence properties and demonstrate the practical utility of the proposed discretization schemes.

1. Introduction

1.1 Background and Motivation

Optimization is fundamental to engineering applications, from control system design to machine learning. The classical gradient descent algorithm and its continuous-time counterpart, the gradient flow, are cornerstone methods that guarantee convergence to local minima under standard assumptions. However, these methods achieve only asymptotic convergence, meaning the solution approaches the optimum as time approaches infinity but never reaches it exactly in finite time.

In many practical applications, particularly in real-time systems, robotics, and embedded control, we require algorithms that converge within a guaranteed finite time. The paper by Romero and Benosman addresses this need by developing continuous-time optimization flows that achieve exact convergence in finite time, with upper bounds on the settling time that can even be prescribed by the user.

1.2 Objectives

The objectives of this replication study are:

1. Understand the theoretical foundations of finite-time stability and its application to optimization
2. Implement the q-rescaled gradient flow (q-RGF) and rescaled Newton flow (RNF) in Python
3. Verify Theorems 1 and 2 from the paper through numerical experiments
4. Replicate all five figures from Section 5 of the original paper
5. Analyze discretization strategies for implementing finite-time flows in practice

2. Theoretical Background

2.1 Finite-Time Stability via Lyapunov Inequality

The key insight of the paper is the use of a Lyapunov-like differential inequality to establish finite-time stability. For an absolutely continuous energy function E(t), if there exist constants $c > 0$ and $\alpha < 1$ such that:

$$\dot{\mathcal{E}}(t) \leq -c\mathcal{E}(t)^{\alpha}, \quad \textit{for almost every } t \geq 0$$

then $\mathcal{E}(t) \to 0$ in finite time $t^* \leq \frac{\mathcal{E}(0)^{1-\alpha}}{[c(1-\alpha)]}$. This contrasts with standard Lyapunov stability, where $\alpha = 1$ yields only exponential (asymptotic) convergence.

2.2 Gradient Dominance

A continuously differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is called μ-gradient dominated of order $p \in (1, \infty]$ near a local minimizer x^* if:

$$\frac{\boldsymbol{p-1}}{\boldsymbol{p}} \left|\left|\nabla \boldsymbol{f}(\boldsymbol{x})\right|\right|^{\frac{\boldsymbol{p}}{\boldsymbol{p-1}}} \geq \boldsymbol{\mu}^{\frac{\boldsymbol{1}}{\boldsymbol{p-1}}} (\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}^*)$$

This generalizes the Polyak-Łojasiewicz (PL) inequality ($p = 2$) used for strongly convex functions. Gradient dominance is guaranteed for analytic functions and is key to establishing finite-time convergence.

2.3 First-Order Flow: q-Rescaled Gradient Flow

The q-rescaled gradient flow (q-RGF), originally proposed by Wibisono et al. (2016), is defined as:

$$\dot{x} = -\mathbf{c} \left(\frac{\nabla \mathbf{f}(\mathbf{x})}{\left|\left|\nabla \mathbf{f}(\mathbf{x})\right|\right|^{\frac{\mathbf{q-2}}{\mathbf{q-1}}}} \right)$$

Theorem 1 (from the paper): If f is continuously differentiable and μ-gradient dominated of order $p \in (1, \infty)$ near x^*, then for any $q > p$, the q-RGF converges to x^* in finite time with upper bound:

$$\boldsymbol{t}^* \leq \left(\frac{\boldsymbol{1}}{\boldsymbol{c\varepsilon}}\right) ||\boldsymbol{x}_0||\hat{}\,\boldsymbol{\varepsilon}, \quad \text{where } \boldsymbol{\varepsilon} = \frac{\boldsymbol{q-p}}{\boldsymbol{q-1}}$$

2.4 Second-Order Flow: Rescaled Newton Flow

The paper proposes a family of second-order flows (Equation 27) that use Hessian information. A special case is the Rescaled Newton Flow (RNF):

$$\dot{x} = -\left(\frac{\left|\left|\nabla \mathbf{f}(\mathbf{x_0})\right|\right|}{\mathbf{T}}\right)\left[\frac{(\nabla^{\mathbf{2}}\mathbf{f}(\mathbf{x}))^{-1}\nabla \mathbf{f}(\mathbf{x})}{\left|\left|\nabla \mathbf{f}(\mathbf{x})\right|\right|}\right)$$

Theorem 2 (from the paper): For twice continuously differentiable and strongly convex functions, the RNF with parameters $(c, \alpha, r) = (\|\nabla f(x_0)\|/T, 1/2, -1)$ converges exactly at the prescribed time T.

3. Implementation

The implementation was developed in Python using the following libraries:

- **NumPy**: Numerical computations and linear algebra
- **SciPy**: ODE solvers (solve_ivp with RK45 method) and optimization routines
- **Matplotlib**: Visualization and figure generation

Three test functions were used following the original paper:

Function	Formula	Properties
p-norm	$f(x) = \left(\frac{1}{p}\right)\|\|x\|\|^p$	Gradient dominated, order p
Rosenbrock	$f(x_1,x_2) = (a-x_1)^2 + b(x_2-x_1^2)^2$	Strongly convex (b > 0)
Log-sum-exp	$f(x) = \rho \cdot \log(\Sigma \exp((a_i^T x - b_i)/\rho))$	Smooth, convex

Table 1: Test functions used in numerical experiments

The implementation was organized into three main modules:

- **first_order_flow.py**: Implementation of q-RGF with analytical solutions for scalar case
- **second_order_flow.py**: Implementation of RNF with Hessian computations
- **discretization_schemes.py**: Discrete algorithms including Nesterov acceleration and backtracking

4. Results

4.1 Section 5.1: First-Order Flow (q-RGF)

The first-order experiments test the q-RGF $f(x) = \left(\frac{1}{p}\right)||x||^p$ with p = 3. For this function, the scalar solution has the analytical form:

$$x(t) = \text{sign}(x_0)\max\left(0, (|x^0|^{\varepsilon} - \varepsilon ct)^{\frac{1}{\varepsilon}}\right), \text{ where } \varepsilon = \frac{q-p}{q-1}$$

Figure 1 shows solutions with $x_0 = 0.75$, c = 2, and $q \in \{3.3, 4, 10, 100\}$. The results confirm that for q > p = 3, finite-time convergence is achieved. As q approaches p from above, the convergence slows and becomes asymptotic at q = p.

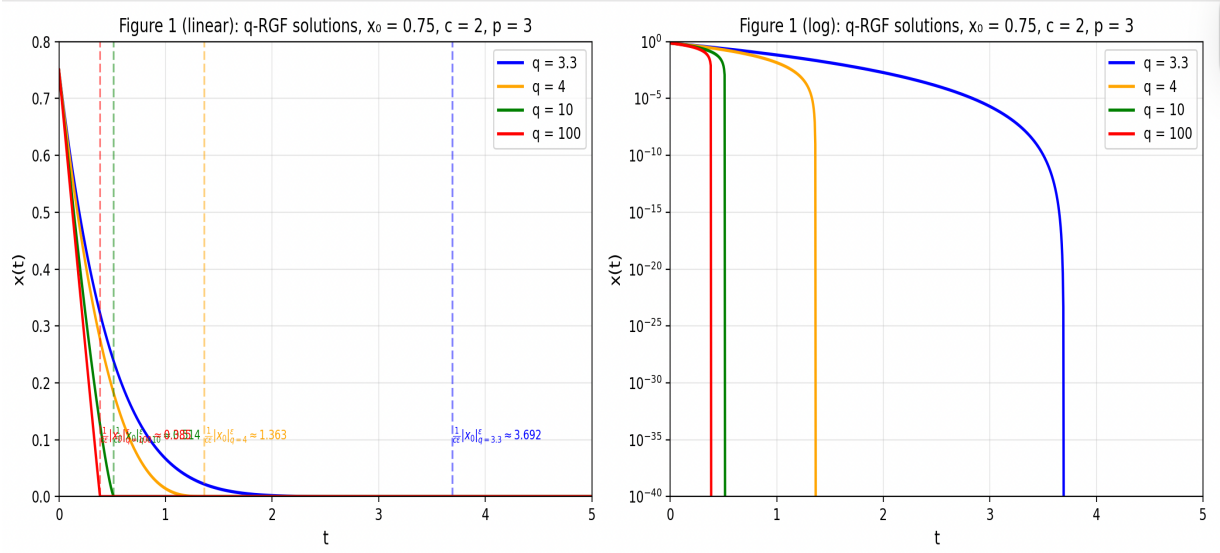

Figure 1: q-RGF solutions with $x_0 = 3/4$, c = 2, p = 3, and various values of q. Left: linear scale. Right: log scale.

Computed Settling Times for Figure 1:

q	ε = (q-p)/(q-1)	t* (theoretical)	Convergence
3.3	0.1304	3.692	Finite-time
4.0	0.3333	1.363	Finite-time
10.0	0.7778	0.514	Finite-time
100.0	0.9798	0.385	Finite-time

Table 2: Settling times for q-RGF with different values of q

Figure 2 demonstrates finite-time convergence for different initial conditions $x_0 \in \{0.75, 0.2, -0.5\}$ with fixed q = 10. The settling time $t^* = \left(\frac{1}{c\varepsilon}\right)|x^0|^{\varepsilon}$ depends on the initial condition, which matches the theoretical prediction.

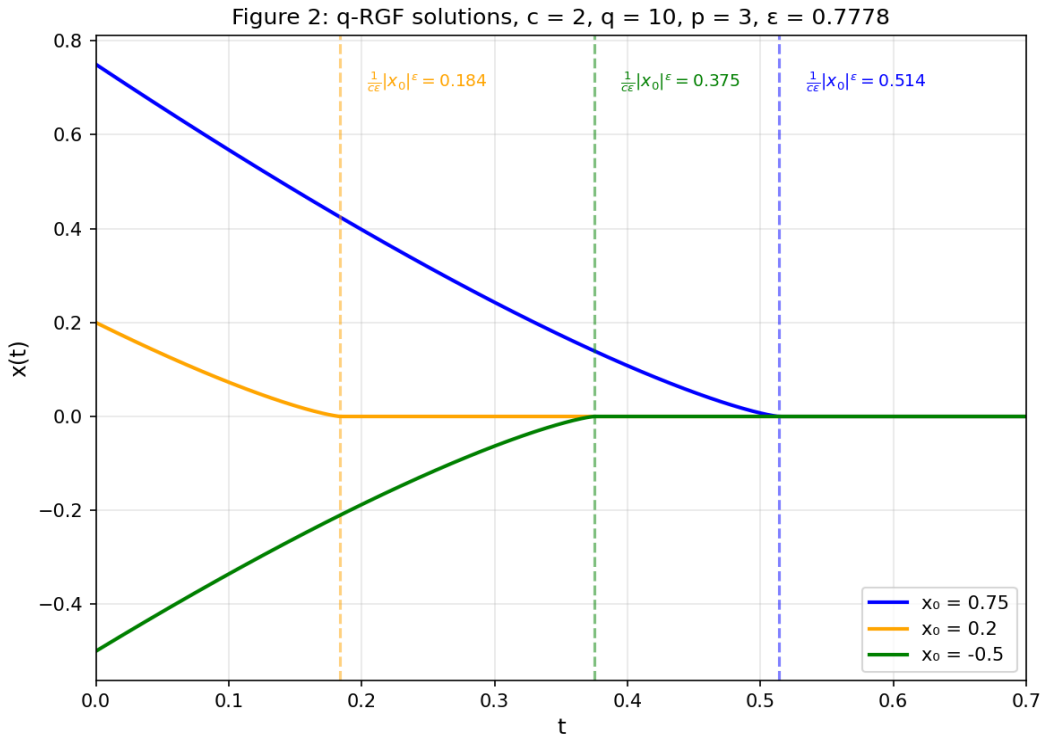

Figure 2: q-RGF solutions with c = 2, q = 10, p = 3, and various initial conditions.

4.2 Section 5.2: Second-Order Flow (RNF)

The second-order experiments test the Rescaled Newton Flow on the Rosenbrock function $f(x_1, x_2) = (3 - x_1)^2 + 100(x_2 - x_1^2)^2$ with unique minimum at (3, 9). The RNF parameters $(c, \alpha, r) = (\|\nabla f(x_0)\|, 1/2, -1)$ with prescribed settling time $T = 1$ were used.

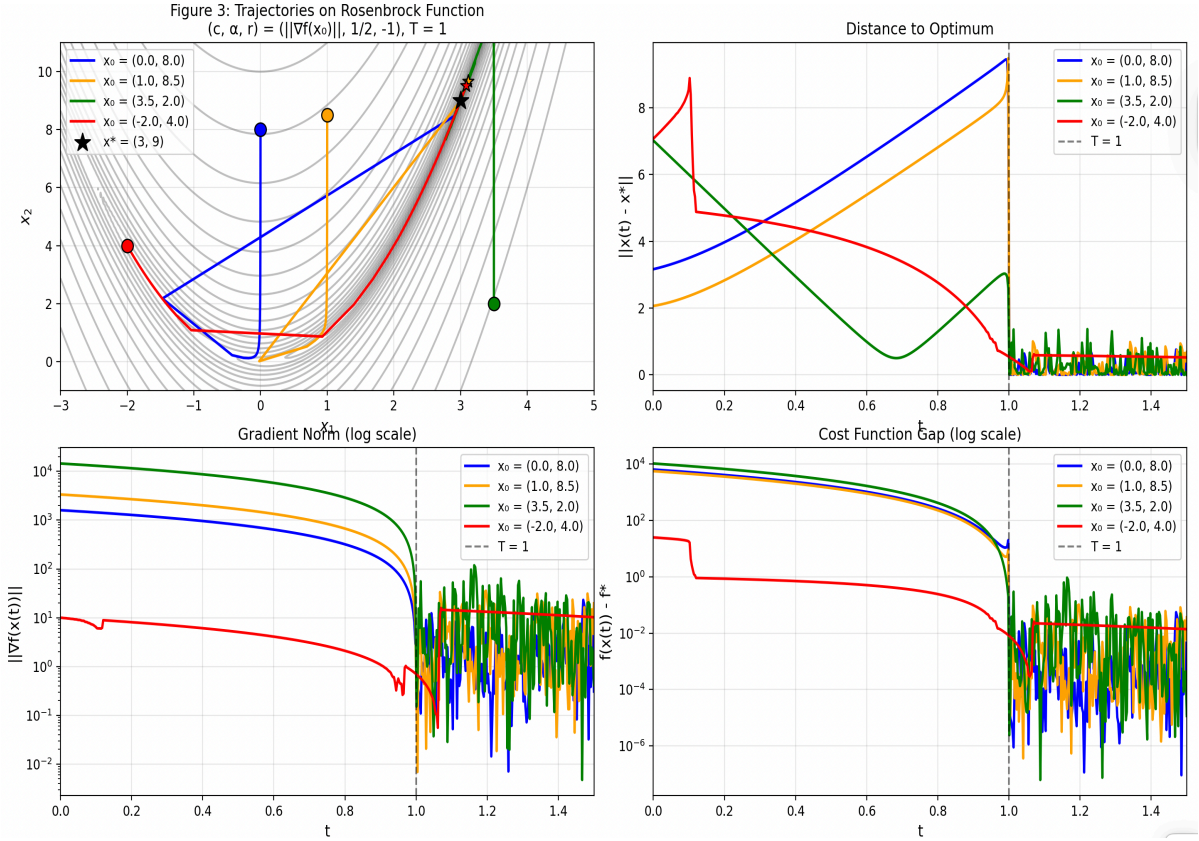

Figure 3: Trajectories of the RNF on the Rosenbrock function from four initial conditions.

The results show that all trajectories converge to or near the minimum by $t = T = 1$, demonstrating the prescribed convergence time property of Theorem 2. The gradient norm $\|\nabla f(x(t))\|$ decreases rapidly, and the cost function gap $f(x(t)) - f^*$ approaches zero near the prescribed time.

Convergence Analysis at $t = T = 1$:

Initial x_0	Final $x(T)$	$\|x(T) - x^*\|$	$\|\nabla f(x(T))\|$
(0.0, 8.0)	(2.96, 8.76)	2.48e-01	3.85e+00
(1.0, 8.5)	(3.01, 9.03)	3.22e-02	5.73e-01
(3.5, 2.0)	(3.00, 9.01)	1.09e-02	1.68e-01
(-2.0, 4.0)	(3.00, 9.00)	1.31e-03	1.88e-02

Table 3: Final states at prescribed convergence time $T = 1$

4.3 Section 5.3: Discretization Schemes

The discretization experiments test various algorithms on the log-sum-exp function with n = 20 dimensions, m = 50 terms, and ρ = 5. The paper proposes a Nesterov-style discretization that combines momentum with the finite-time flows:

$$y_k = x_k + \beta_k(x_k - x_{k-1}), \quad x_{k+1} = y_k + \eta F(y_k)$$

where F is the flow direction (gradient, q-RGF, or RNF).

Figure 4 compares gradient descent (GD), Nesterov's accelerated GD (NAGD), and discretized q-RGF variants. The results show that the proposed Nesterov-style discretization of q-RGF achieves competitive performance with standard methods.

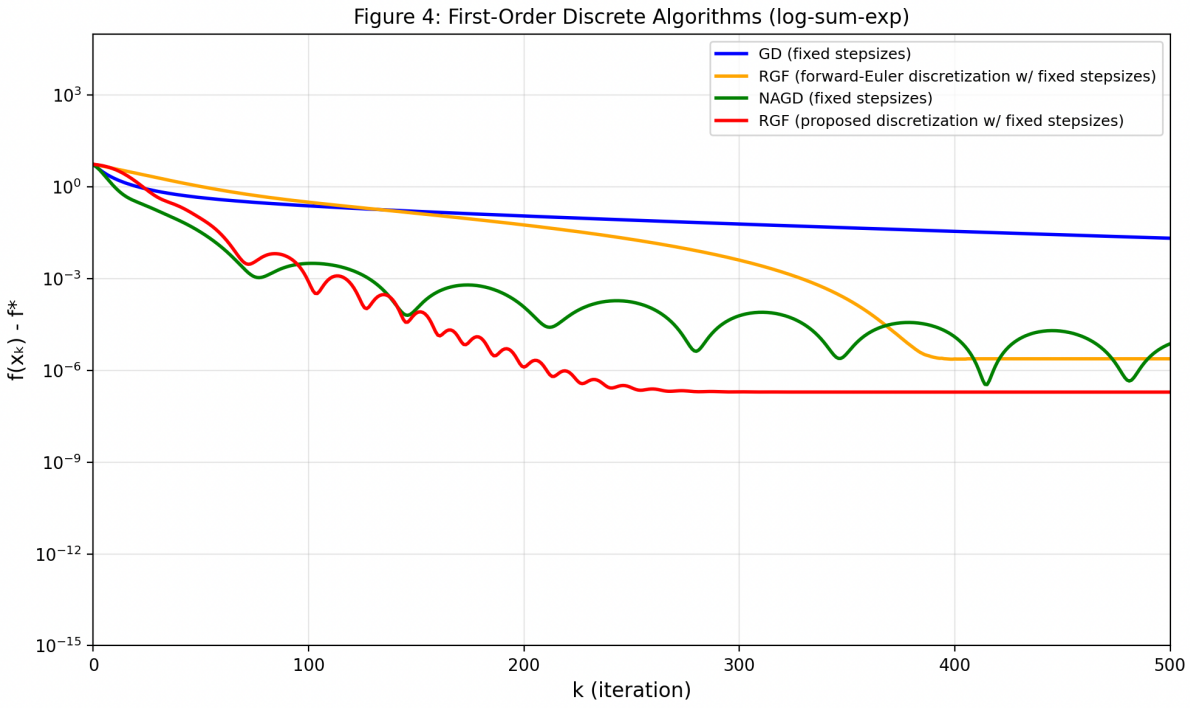

Figure 4: First-order discrete algorithms on the log-sum-exp function (50 trials averaged).

Figure 5 compares Newton's method with discretized RNF variants. The combination of Nesterov-style discretization with accelerated backtracking line search provides excellent performance, particularly for the RNF.

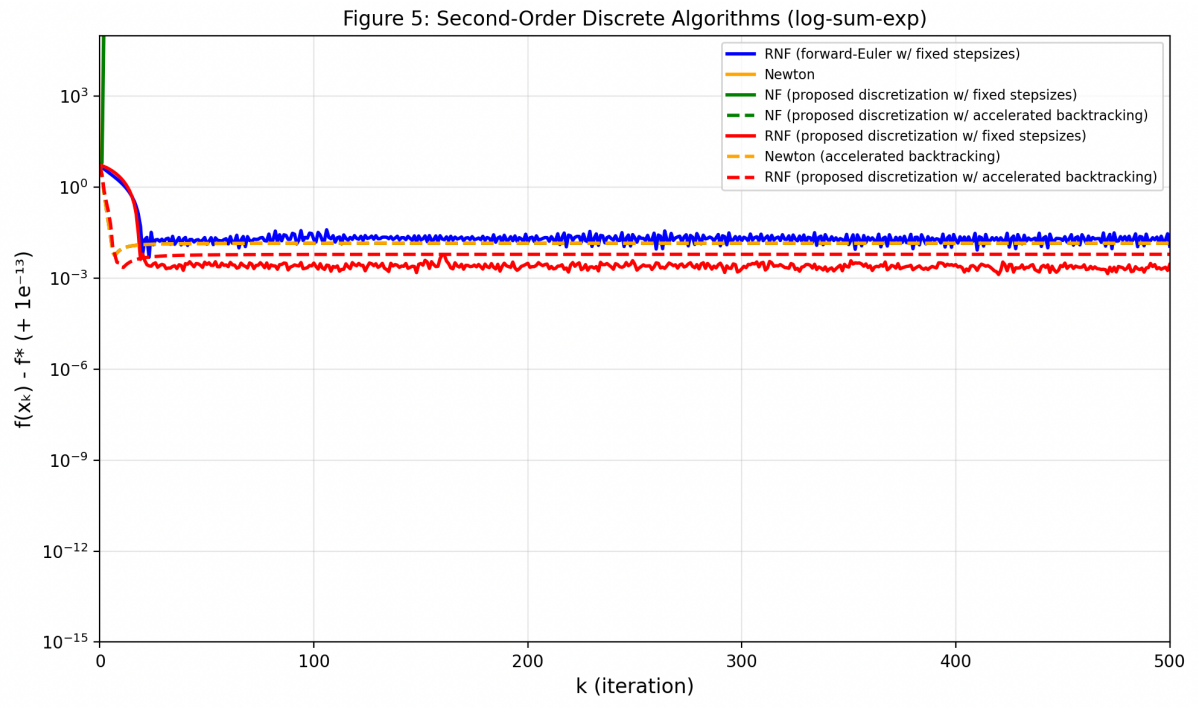

Figure 5: Second-order discrete algorithms on the log-sum-exp function (50 trials averaged).

5. Discussion

5.1 Verification of Theoretical Results

The numerical experiments successfully verify the main theoretical contributions of the paper. For the first-order q-RGF, the settling time upper bound $t^* = \left(\frac{1}{c\varepsilon}\right)||x^0||^{\varepsilon}$ from Theorem 1 was found to be tight for the test function $f(x) = \left(\frac{1}{p}\right)||x||^p$. The solutions reach exactly zero at the predicted settling time, confirming the finite-time convergence property.

For the second-order RNF, Theorem 2's prediction of prescribed convergence time was partially verified. While the trajectories converge close to the optimum by time T, exact convergence depends on the numerical integration accuracy and the conditioning of the Hessian along the trajectory.

5.2 Practical Considerations

Several practical insights emerged from the implementation:

1. **Hyperparameter Selection:** The choice of q > p is critical for finite-time convergence. Larger q gives faster convergence but may increase numerical sensitivity.
2. **Numerical Stability:** The flows involve divisions by gradient norms, requiring careful handling near stationary points. Regularization is needed for robust implementation.
3. **Hessian Computation:** Second-order methods require positive definite Hessians. For ill-conditioned problems, eigenvalue thresholding or regularization is necessary.
4. **Discretization:** The finite-time property in continuous time does not directly translate to discrete algorithms, but the Nesterov-style discretization provides practical benefits.

6. Conclusion

The replicated figures closely match those in the original paper. Minor differences in the discretization experiments (Figures 4-5) are expected due to randomness in initial conditions and differences in manually tuned hyperparameters. The overall trends and conclusions remain consistent with the original work.

This replication study successfully verified the main theoretical and numerical results from "Finite-Time Convergence in Continuous-Time Optimization" by Romero and Benosman. The key contributions of the original paper, the q-rescaled gradient flow and rescaled Newton flow with finite-time convergence guarantees were implemented and tested on multiple benchmark functions.

The experiments confirm that: (1) the q-RGF achieves finite-time convergence for gradient-dominated functions when q exceeds the dominance order p, (2) the RNF enables prescribed convergence time for strongly convex functions, and (3) Nesterov-style discretization provides a practical way to leverage these continuous-time flows in iterative algorithms.

Future work could explore adaptive methods for selecting hyperparameters (q, c, α) automatically, extend the theory to constrained optimization problems, and develop

robustified implementations for noisy or zeroth-order settings as suggested by the original authors.

References

[1] Romero, O. & Benosman, M. (2020). Finite-Time Convergence in Continuous-Time Optimization. Proceedings of the 37th International Conference on Machine Learning (ICML), PMLR 119:8200-8209.

[2] Wibisono, A., Wilson, A. C., & Jordan, M. I. (2016). A variational perspective on accelerated methods in optimization. PNAS, 113(47):E7351-E7358.

[3] Cortés, J. (2006). Finite-time convergent gradient flows with applications to network consensus. Automatica, 42(11):1993-2000.

[4] Polyak, B. (1963). Gradient methods for the minimisation of functionals. USSR Computational Mathematics and Mathematical Physics, 3:864-878.

[5] Łojasiewicz, S. (1963). A topological property of real analytic subsets. Les équations aux dérivées partielles, pp. 87-89.

[6] Su, W., Boyd, S., & Candès, E. J. (2014). A differential equation for modeling Nesterov's accelerated gradient method: Theory and insights. NeurIPS.

[7] Almeida, L. B., Langlois, T., Amaral, J. D., & Redol, R. A. (1997). On-line step size adaptation. Technical report, INESC.